PROCEDURES AND TEMPLATES FOR TEST EXECUTION

Release Date: November 16, 2000

Contract Number: GS-35F-4919H
Order Number: T0000AJ3739

CLIN #010-1

Prepared for:

U. S. DEPARTMENT OF EDUCATION
OFFICE OF STUDENT FINANCIAL ASSISTANCE (SFA)
400 Maryland Avenue, SW
Washington, DC 20202

Prepared by:

CTGi
Suite 250
10461 White Granite Drive
Oakton, VA 22124

This page left intentionally left blank

FOREWORD

CTG, Incorporated (CTGi) would like to thank the following personnel whose dedication and involvement made the development and completion of this document possible.

Kelly Conboy

Candace Jones

Frank Majewski

Annette Mazie

Antony Mosley

Isaac Sanvee

Paul Stocks

Sandra Stocks
Director of Operations,
Washington Metropolitan Area
CTGi

This page left intentionally left blank

EXECUTIVE SUMMARY

This document has been prepared for the United States (U. S.) Department of Education, Office of Student Financial Assistance (SFA), for the purpose of developing a standardized procedure for the execution and evaluation of software system tests and error correction arising from testing.

This document, along with those listed below, will be integrated into the U. S. Department of Education, SFA, System Integration and Testing (SI&T) Process Handbook. The U. S. Department of Education, SFA, SI&T Process Handbook will then become integrated into the overall U. S. Department of Education, SFA, Modernization Technology Handbook. The remaining documents that will comprise the U. S. Department of Education, SFA, SI&T Process Handbook include:

- System Integration and Testing Standards
- Test Performance Measurements
- Procedures and Templates for Creating Test Conditions, Test Scenarios, and Testing Data
- Procedures and Templates for System Configuration Management (CM) and Quality Control (QC)
- Procedures For Using Testing Tool

The above listed procedures, templates, and guides were prepared and delivered as separate documents.

All SI&T guidelines and procedures are focused on the support, development, and execution of a comprehensive SI&T processes. To this end, this document contains information on understanding issues related to test planning roles, responsibilities of the organization, and required testing practices. The standards and procedures used in the support of this document are reflective of industry best practices, practices of other federal government agencies, and various governing standards and literature regarding the integration and testing process.

This page left intentionally left blank

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont'd)

TABLE OF CONTENTS(Cont'd)

This page left intentionally left blank

LIST OF FIGURES

This page left intentionally left blank

LIST OF ACRONYMS

CM	Configuration Management
COTS	Commercial Off the Shelf
CSCI	Computer Software Configuration Item
DID	Data Item Description
HWCI	Hardware Configuration Item
IRS	Interface Requirements Specification
QA	Quality Assurance
QC	Quality Control
SDF	Software Development File
SDP	Software Development Plan
SRD	Software Requirements Document
SRS	Software Requirements Specification
SU	Software Unit
SFA	Office of Student Financial Assistance
SI&T	System Integration and Testing
SPR	System Problem Report
SQT	System Qualification Test
SSS	System or Subsystem Specifications
STD	System Test Description
STP	System Test Plan
STR	System Test Report
TRR	Test Readiness Review

This page left intentionally left blank

1. INTRODUCTION

1.1 Background

The U. S. Department of Education, Office of Student Financial Assistance (SFA), contracted CTG, Incorporated (CTGi), in August 2000, to develop standardized System Integration and Test (SI&T) procedures. These procedures will be used for guidance, planning, and implementation involving current and future U. S. Department of Education, SFA, enterprise information technology systems projects.

1.2 Objective

The objective of this document is to provide procedures and templates for standardization of the development of test execution, test evaluation and test error correction necessary in preparation for the implementation of all types of U. S. Department of Education information technology SI&T.

The creation of standardized procedures for the execution of test execution, test evaluation and test error correction is critical to the establishing a consistent and comprehensive testing capability. These standard procedures will be documented and available to all personnel involved in the oversight and testing of software applications. The purpose of these procedures is to promote a consistent approach in both development and test of application products. The standard integration and testing processes include the essentials of what has to be done, why, and who performs and completes the work. Procedural activities are provided to explain how to complete the process (the sequence of tasks or task steps that have to be performed), when the work is to be performed and completed, and criteria for measuring the quality of the work.

1.3 Applicability

When the SI&T process is performed by either the U. S. Department of Education, SFA, staff and/or contractors this document will apply, unless specifically excluded, in the program/project plan, contract, etc. This document will be used for the creation of guidelines and procedures for the planning, preparation, execution, analysis, and evaluation, of all types of U. S. Department of Education, SFA, information technology project integration and testing.

1.4 Document Organization

This document contains nine narrative sections, a Glossary, a Bibliography, and four appendices. Section 1 provides brief background information and states the guiding objective and applicability. Section 2 provides an overview of test execution phases and conditions. Section 3 provides the necessary procedures for the development and conducting a Software Unit (SU) Test. Section 4 provides information for conducting the Testing Test Readiness Review (TRR) for the Integration Test phase. Section 5 covers both Computer Software Configuration Item and Regression Tests. Section 6 provides information for conducting the TRR for Performance

Testing. Section 7 provides procedures for conducting Performance Testing. Section 8 provides information for conducting the System Qualification Test (SQT) TRR. Section 9 provides procedures for conducting the SQT. Appendix A proves an overview of testing techniques that will be employed during the SI&T process. Appendix B provides an example of a template for creating a TRR checklist. Appendix C provides the template for creating a System Test Report (STR). Appendix D provides the template for creating a System Problem Report (SPR).

2. TESTING OVERVIEW AND RELATIONSHIPS

The overall objective of a testing methodology is to ensure that new or upgraded systems are production ready, and are ready to be used in a production environment. The goal of any methodology is to accomplish these objectives in a reasonable amount of time. Figure 2.1, summarizes the objective and lists an example for each testing type of the SI&T strategy.

Stage	Objectives
Software Unit Test	• Thoroughly test the SU by focusing on all possible test conditions. • Test the functionality and technical components within the confines of a single SU of work. • One SU test is performed for each SU that is written or modified. • Integrate SUs into approved Computer Software Configuration Item (CSCI) modules.
Integration Test	• Test the application/system in a simulated production environment. • Integrate and test approved CSCI modules. • Conduct Regression Test, if CSCI modules are integrated into a production system. • Regression Test conducted using plans created for SU or previous integration and/or performance testing. • Perform testing according to System Test Description (STD) and test case(s). • Test the input(s) and output(s) to any interfaced systems.
Performance Test	• Complete an environment test to ensure communications with other systems work correctly. • Test the application/system in a simulated production environment. • System interface and data exchange test. • Model test system environment that will enable processing of predicted production volumes, both batch and on-line. • Ensure that response times for the system or application meet Software Requirements Document (SRD) specifications.
System Qualification Test	• Test software or system in a simulated production environment with other release applications and current production applications using business test case(s) that integrate systems and work flow. • Verify software or system against requirements defined in the SRD in an integrated testing environment. • Ensure users verify that processing meets requirements.

Figure 2.1 SI&T Testing Phases

2.1 Determining System Test Execution Conditions

Activities associated with developing a comprehensive testing execution and evaluation approach are presented in Figure 2.2.

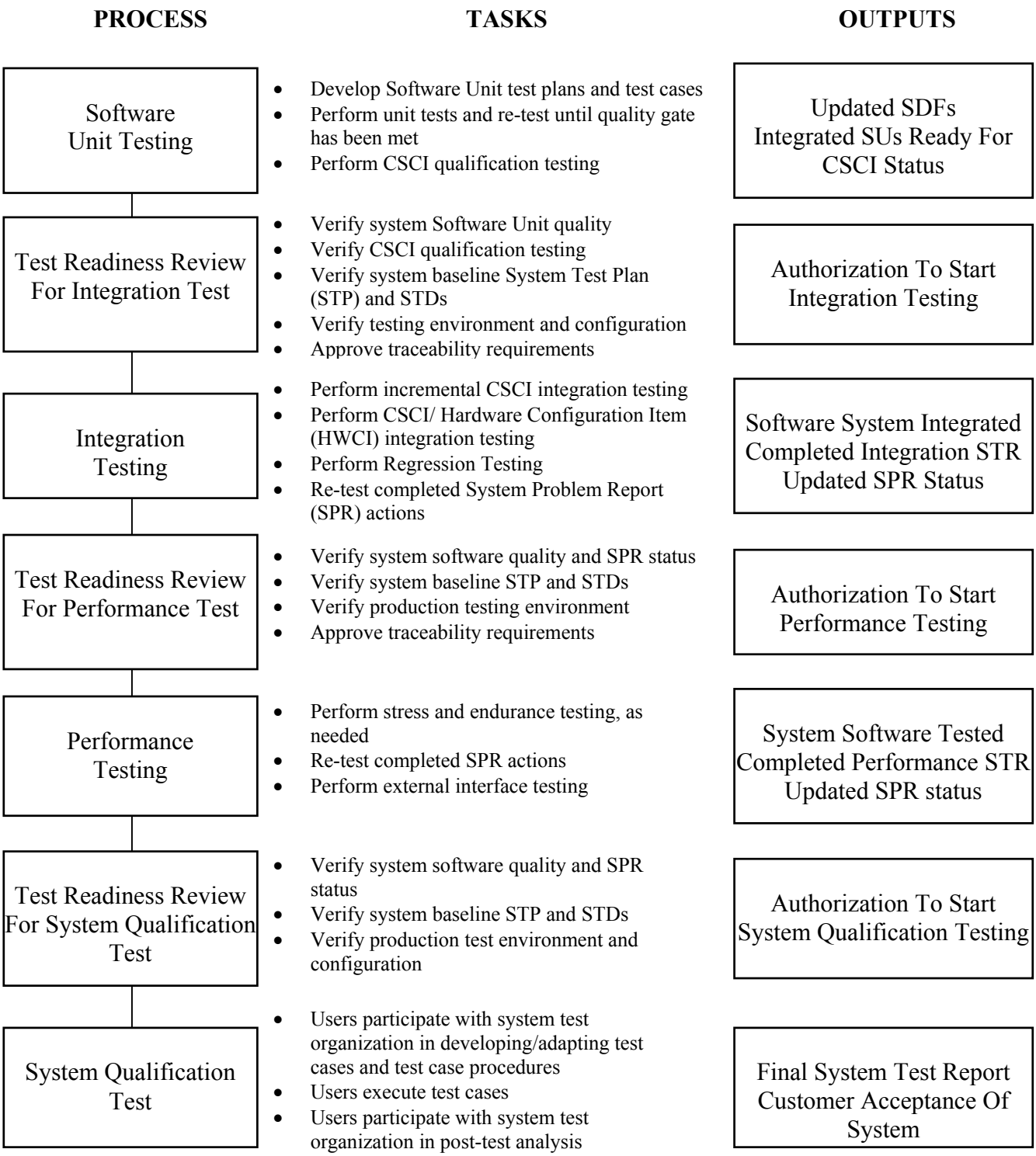

Figure 2.2. Procedures For Test Execution and Evaluation

3. PROCEDURES FOR SOFTWARE UNIT TESTING

The objective of SU testing is to identify and correct internal logic errors contained in a SU. To meet this objective, the software development group performs selected path testing within each SU and exercises at least 80 percent of the affected branch conditions in all possible directions at least once and every affected line of code at least once. SU test drivers and stubs will be developed, as needed, and placed under configuration control as part of the Software Development File (SDF). In addition, results of each SU test are recorded in the SDF.

PROCESS	TASKS	OUTPUTS
Software Unit Testing	• Develop Software Unit test plans and test cases • Perform unit tests and re-test until quality gate has been met • Perform CSCI qualification testing	Updated SDFs Integrated SUs ready for CSCI status

A diagram of the SU Test phase of the SI&T process is shown in Figure 3.1.

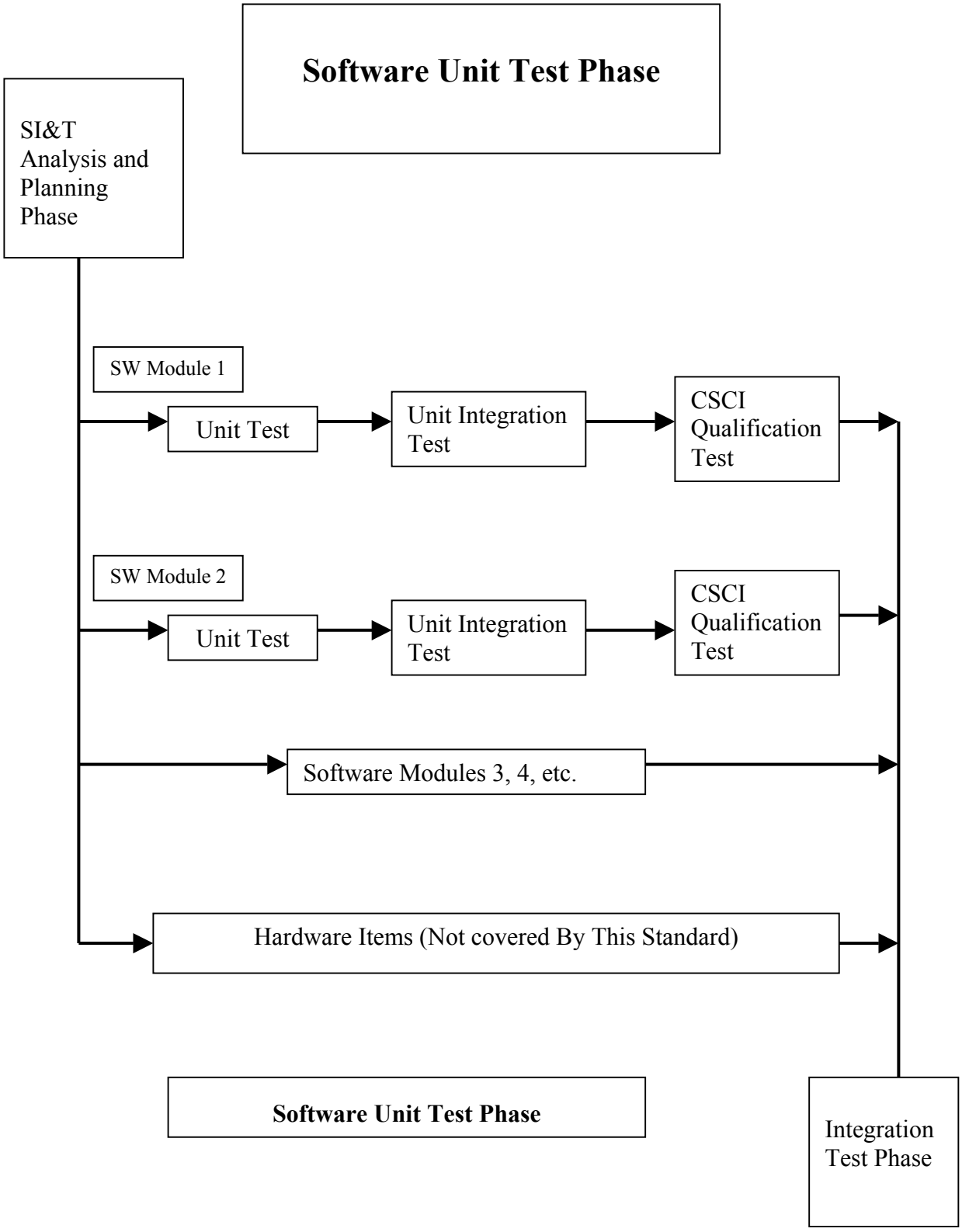

Figure 3.1 Software Unit Test Phase

3.1 Procedures for Software Unit Test Development

A comprehensive set of repeatable SU test cases and test case procedures will be developed to ensure the correctness of each SU.

3.1.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. Participants will be responsible for overseeing the activities performed and provide documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the test group will vary from system to system due to system complexity, size, function, etc.

3.1.2 Entrance Criteria

a. Software requirements have been allocated to individual SUs and documented in the appropriate SDF.
b. The design of the SU to be tested is complete and documented in the appropriate SDF.

3.1.3 Inputs

Allocated requirements and approved design of the SU to be tested.

3.1.4 Activities

a. Review the STP to determine the types of test cases required for the SU.
b. Review all inputs and outputs for the SU. Create test data necessary to stimulate unit execution, provide inputs, and capture final outputs.
c. Identify paths that may be taken by SU. Use a static analysis tool and/or a peer review to determine SU paths.
d. Identify data variables and conditions that determine which paths are taken.
e. Identify parameter limits for all input, output, and control parameters.
f. Examine and list all error handling facilities of SU and error conditions.
g. Identify a set of test case procedures to conduct the required SU testing (path, boundary condition, input validation, and syntax). There should be a test case

procedure for each path through the SU to test the upper and lower limits of all parameters, and to test all error conditions (pass and fail).

h. Write step-by-step instructions and/or test data necessary for each test case procedure.

i. Document the SU test plan in the SDF.

j. Submit the SU test plan for peer review, and resolve all comments.

3.1.5 Outputs

a. Approved SU test plan documented in appropriate SDF.

b. Completed test cases and test case procedures.

c. Completed test data.

3.1.6 Exit Criteria

SDFs that contain test plans for all included SUs.

3.2 Procedures For Software Unit Test Execution

SU testing will cover the following areas:

- Path Testing - Execution of every logic branch and line of code to find logic errors in control structures, dead code, errors at loop boundaries, and errors in loop initializations. This includes every state and every mode.
- Boundary Condition Testing - To find errors in input and output parameter tolerances and to verify that program limits are correctly stated and implemented.

3.2.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. Participants will be responsible for overseeing the activities to be performed and providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the participants will vary from system to system due to system complexity, size, function, etc.

3.2.2 Entrance Criteria

a. Approved SU test cases and test case procedures.
b. SU source code developed and peer reviewed.

3.2.3 Inputs

a. Approved SU test cases, test case procedures, and test data.
b. The SU source code developed and ready for unit test.

3.2.4 Activities

a. Review SU test cases.
b. Execute each SU test case procedure.
c. Compare test results with expected results.
d. Document the results in an SDF. This report should contain all data recorded from executed test cases and deviations from the test case procedures.
e. If the unit did not successfully pass the SU test case, schedule the SU for rework.
f. Upon successful completion of the SU test cases, SU is eligible for integration testing.

3.2.5 Outputs

a. An updated SDF reflecting SU test results.
b. Successfully tested SUs.

3.2.6 Exit Criteria

a. An updated SDF reflecting SU testing results.
b. Successfully tested SUs.

3.3 Procedures For Software Unit Integration Testing

At this level, SUs are incrementally integrated to form larger and more complex software modules (i.e., CSCI modules). Integration activities continue until all needed SUs are integrated into functioning CSCI modules defined in the STP.

3.3.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. Participants will be responsible for overseeing the activities to be performed and providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the participants will vary from system to system due to system complexity, size, system, etc.

NOTE: *The person(s) responsible for fulfilling the requirements in this function should not be the persons who performed detailed design or implementation of software in the system. This does not preclude persons who performed detailed design or implementation of software in the system from contributing to the process, for example, by contributing test cases that rely on knowledge of the system internal implementation.*

3.3.2 Entrance Criteria

a. Test resources in place and ready for operation.
b. SUs that have successfully completed testing.
c. Approved SU test cases, test case procedures, and test data.

3.3.3 Inputs

a. Updated SDF reflecting SU test results.
b. Baseline STP.
c. STD, if needed for reference.
d. Test-ready SUs.
e. Test cases, test case procedures, and test data.

3.3.4 Activities

a. Upon receipt of a request, a build is developed incorporating the cited SUs.
b. Integrate successfully tested SU into CSCIs.
c. Update any existing test data in response to approved software changes/fixes.
d. Conduct testing in accordance with test case procedures. Record test results as they are observed.

e. Perform required post-test analysis or data reduction to determine pass/fail criteria, as specified in the test case procedures.

f. Compare test results with expected results.

g. Document all test results in a testing report. A report will be created that contains a summary of all data recorded from executed test case results with note(s) of deviations from test case procedures.

h. Document any problems detected in SDF.

i. Repeat or revise all erroneous test cases as documented in the SDF until quality gate is achieved.

j. File the SDF and submit a copy of the SDF for review, analysis, and approval.

3.3.5 Outputs

a. A version-specific test results report.

b. An updated SDF reflecting integrated SU tests.

3.3.6 Exit Criteria

a. Integration test report reviewed and approved.

b. A SDF that documents all detected problems.

c. Integrated CSCI modules ready for CSCI qualification testing.

3.4 Procedures For Computer Software Configuration Item Qualification Testing

After completion of unit integration testing, a group will conduct a CSCI qualification test on each fully integrated component (i.e., CSCI). The purpose of CSCI qualification testing is to verify satisfaction of CSCI requirements as documented in the SRD.

CSCI qualification testing will cover the following areas:

- Path Testing - Execution of every logic branch and line of code to find logic errors in control structures, dead code, errors at loop boundaries, and errors in loop initializations. This includes every state and every mode.
- Boundary Condition Testing - To find errors in input and output parameter tolerances and verify that the program limits are correctly stated and implemented.

3.4.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. Participants will be responsible for overseeing the activities

to be performed and providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the participants will vary from system to system due to system complexity, size, function, etc.

NOTE: *The person(s) responsible for fulfilling the requirements in this function shall not be the persons who performed detailed design or implementation of software in the system. This does not preclude persons who performed detailed design or implementation of software in the system from contributing to the process, for example, by contributing test cases that rely on knowledge of the system internal implementation.*

3.4.2 Entrance Criteria

a. A verified testing environment.
b. CSCI qualification test cases prepared.
c. Integrated CSCI ready for qualification test.

3.4.3 Inputs

a. Updated SDF reflecting SU testing and CSCI integration results.
b. Test case, test case procedures, testing data, and expected results data.

3.4.4 Activities

a. Perform CSCI qualification testing.
b. Place successfully tested CSCI software under CM.
c. Use test cases that exercise the CSCI through all requirements.
d. The test team meets prior to scheduled test to brief participants on roles and verify readiness of test configuration and materials.
e. Conduct pre-test inspections of test hardware configurations and interfaces to external systems.
f. Load and initialize CSCI software to meet prescribed test conditions.
g. Execute test, following prescribed test cases. Record results on operator logs and automated recording media.
h. Upon completion of test session, debrief assigned test personnel on test observations.

i. Submit SPRs, as required.
j. Submit CSCI qualification test report for approval.

3.4.5 Outputs

a. An updated SDF reflecting CSCI qualification testing.
b. Tested CSCI under CM.
c. Automated test logs, if applicable.

3.4.6 Exit Criteria

a. Approved SDF for all SU and CSCI activities.
b. Approved CSCI ready for Integration Testing.

This page left intentionally left blank

4. TEST READINESS REVIEW FOR INTEGRATION TEST

Before progressing to Integration Testing, a review of the individual components and/or activities for the test will be conducted by project management in the form of a Test Readiness Review (TRR).

PROCESS	TASKS	OUTPUTS
Test Readiness Review	• Verify system software quality • Verify baseline STP and STDs • Verify system test environment • Approve testing plan • Approve traceability requirements	Authorization To Start Integration Test

4.1 Test Readiness Review Checklist

A TRR checklist is recommended for guidance in the assessment of actions and documentation needed to ascertain the status of a system prior to entering the next stage of the SI&T process. Actions and items completed and approved will be noted, and the date of completion and approval will be entered on the checklist. Actions and items needing attention will be noted. At the conclusion of the TRR, the checklist will be copied and distributed to all participants. Project management and Quality Assurance (QA) will have final authority for proceeding into the Integration Test phase.

See Appendix B for the TRR Checklist template.

4.2 Test Readiness Review Meeting

The purpose of the TRR meeting is to ensure the quality and status of the system or software prior to its release for Integration Testing. The TRR will be conducted a minimum of two business days before Integration Testing begins.

4.2.1 Participants

a. Project Manager
b. Software development engineers.
c. System test engineers.
d. QA engineers.
e. CM engineers.
f. System network engineers (if necessary).

4.2.2 Entrance Criteria

System has completed the SU Test phase of the SI&T process.

4.2.3 Inputs

a. Baseline STP and STD(s).
b. SU SDF(s).
c. Testing plan for Integration Test.
d. Test cases, test case procedures, and test data for Integration Test.
e. Production version of system or software for Regression Test, if necessary.

4.2.4 Activities

a. Preparation of test schedule.
b. Approval of a test plan.
c. Approval of test cases, test case procedures, and test data.
d. Ascertain status of system.
e. Approval of traceability from SRD to test cases.
f. Establish that HWCI is under CM procedures.

4.2.5 Outputs

a. Approval to proceed to Integration Test phase.
b. Decision to withhold approval of system for Integration Test phase.

4.2.6 Exit Criteria

Decision regarding the status of system to proceed to Integration Test phase.

5. PROCEDURES FOR INTEGRATION TESTING

An Integration Test can be either an integration of various approved CSCI modules leading to the creation of a new system, or it can be for one or more approved CSCI modules being integrated into a production system for an upgrade or modification. If CSCI module(s) are being used to upgrade or modify a production system, additional action must be taken by performing a Regression Test.

If the Integration Test phase is for the creation of a new system, the approved CSCIs necessary to satisfy all functions stated in the Software Requirements Document (SRD) are integrated to create the system. The test team will test and verify the compatibility of each new CSCI with the previously integrated CSCIs. This process will continue until all needed CSCIs are integrated in to the system.

When a previously released system, which is being modified or upgraded by the addition of CSCI(s), both the CSCI and the previously released system will undergo Regression Testing.

Regression Testing is selective re-testing of the modified or upgraded system to detect faults introduced or caused by the modification or upgrades. Regression Testing verifies that the modified or upgraded system continues to operate and function to original requirements and specifications.

SPRs reported as resolved will be re-tested. The results of the Integration Test will be documented in the STR.

PROCESS	TASKS	OUTPUTS
Integration Testing	• Perform incremental CSCI integration testing • Perform Regression testing • Re-test SPRs reported by development as Resolved. • System Test Report (STR) created	System Software Approved For Performance Testing

5.1 Computer Software Configuration Item Integration Testing

During CSCI Integration Testing, CSCIs are incrementally added to previously tested CSCI modules to form larger and more complex system function and actions. Detailed CSCI Integration Test cases are documented in the STD.

5.1.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. Participants will be responsible for overseeing the activities to be performed and providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the participants will vary from system to system due to system complexity, system, system, etc.

NOTE: *The person(s) responsible for fulfilling the requirements in this function should not be the persons who performed detailed design or implementation of software in the system. This does not preclude persons who performed detailed design or implementation of software in the system from contributing to the process, for example, by contributing test cases that rely on knowledge of the system internal implementation.*

5.1.2 Entrance Criteria

a. Software has successfully completed SU testing.
b. CSCIs under CM.
c. Test cases have been developed and are presented in the STDs.
d. Approval of TRR for Integration Testing.
e. Production version of system that is being upgraded or modified, if necessary.

5.1.3 Inputs

a. STD.
b. Approved test cases, test case procedures, and test data.
c. Upgraded SDFs for reference, if needed.
d. Test qualified CSCIs.

5.1.4 Activities

a. CSCIs are integrated in the test environment.
b. Update any existing test data in response to approved test case changes.
c. Conduct test case procedures in accordance with the test case. Record test results as they are observed.
d. Perform required post-test analysis or data reduction to determine pass/fail criteria as specified in the test case procedures.
e. Compare actual test results with expected results.
f. Document all test results of the Integrated Test, using the STR. The report should contain a summary of all data recorded from executed test case results. Note deviations from the test case.

g. Document any problems detected on SPR form(s) and submit SPR(s) for review and analysis.

h. Repeat or revise all erroneous test cases documented in the SPR until quality gate is achieved.

i. Submit a copy of the STR for review, analysis, and approval.

5.1.5 Outputs

a. An Integration Test STR.

b. SPRs submitted for all problems detected in the integrated CSCIs.

5.1.6 Exit Criteria

a. Integration STR reviewed and approved.

b. SPRs submitted for all detected problems.

c. Integrated CSCIs ready for Regression Testing or Performance Testing.

5.2 Regression Testing

The STD for this test will be developed from previously executed test cases covering mission-critical functions identified in the SRD. Specific mission-critical functions are chosen for re-testing to ensure overall operational effectiveness of the system. Modifications to Regression Test cases will be at the discretion of the system testing organization and QA.

5.2.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. Participants will be responsible for overseeing the activities to be performed and providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the participants will vary from system to system due to system complexity, size, function, etc.

NOTE: *The person(s) responsible for fulfilling the requirements in this function should not be the persons who performed detailed design or implementation of software in the system. This does not preclude persons who performed detailed design or implementation of software in the*

system from contributing to the process, for example, by contributing test cases that rely on knowledge of the system internal implementation.

5.2.2 Entrance Criteria

a. A production version of the system undergoing Regression Testing.
b. Subject software is under CM.
c. Test cases have been selected and modified for Regression Testing.
d. Authorization to proceed obtained through TRR for Integration Testing.

5.2.3 Inputs

A production version of the system undergoing Regression Testing.

5.2.4 Activities

a. Conduct test cases procedures in accordance with the test case. Record test results as they are observed.
b. Compare actual test results with expected results.
c. Document problems detected on SPR form and submit SPR for review and analysis.
d. Repeat or revise all erroneous test cases as documented in the SPR.
e. File test report and submit a copy of test report for review, analysis, and approval.
f. Document all test results using the Regression STR. This report will contain a summary of all data recorded from executed test case results and deviations from test case procedures.

5.2.5 Outputs

a. A version-specific STR that includes both CSCI Integration and Regression Testing.
b. SPRs submitted for all detected problems.

5.2.6 Exit Criteria

a. Regression Test STR reviewed and approved.
b. SPRs submitted for all detected problems.
c. Integrated system ready for Performance Test phase.

6. TEST READINESS REVIEW FOR PERFORMANCE TEST

Before progressing to Performance Testing, a review of the individual components and/or activities necessary for the test will be conducted by project management in the form of a TRR.

PROCESS	TASKS	OUTPUTS
Test Readiness Review	• Verify system software quality and SPR status • Verify baseline STP and STDs • Verify system test environment • Approve testing plan	Authorization to Start Performance Test

6.1 Test Readiness Review Checklist

A TRR checklist is recommended for guidance in the assessment of actions and documentation needed to ascertain the status of a system prior to entering the next stage of the SI&T process. Actions and items completed and approved will be noted, and the date of completion and approval will be entered on the checklist. Actions and items needing attention will also be noted. At the conclusion of the TRR, the checklist will be copied and distributed to all participants. Project management and QA will have final authority for proceeding into the Performance Test phase.

See Appendix B for the TRR Checklist template.

6.2 Test Readiness Review

The purpose of the TRR meeting is to ensure the quality and status of the system or software prior to its release for Performance Testing. The TRR will be conducted at a minimum of two business days before Performance Testing begins.

6.2.1 Participants

a. Project Manager
b. Software development engineers.
c. System test engineers.
d. QA engineers.
e. CM engineers.
f. System network engineers (if necessary).

6.2.2 Entrance Criteria

The system has completed the Integration Test phase of the SI&T process.

6.2.3 Inputs

a. Baseline STP and STD(s).
b. Performance Test STR.
c. Testing plan for Performance Testing.
d. Test cases, test case procedures, and test data for Performance Testing.
e. System SPR status report.

6.2.4 Activities

a. Preparation of test schedule.
b. Approval of a test plan.
c. Approval of test cases, test case procedures, and test data.
d. Ascertain status of system.
e. Review of SPR status.
f. Approval of traceability from SRD to test cases.
g. Establish that HWCI is under CM procedures.

6.2.5 Outputs

a. Approval to proceed to Performance Test phase.
b. Decision to withhold approval of system for Performance Test phase.

6.2.6 Exit Criteria

Decision regarding the status of the system to proceed to Performance Test phase.

7. PROCEDURES FOR PERFORMANCE TEST

Upon completion of Integration Testing, the system is ready for Performance Testing. This phase of testing will be conducted in accordance with the STP and the applicable STD(s). The purpose of Performance Testing is to validate the compatibility of all hardware and software components in accordance with the STP. The Performance Test will be conducted in an environment simulating real world usage.

PROCESS	ACTIONS	OUTPUTS
Performance Testing	• Perform stress and endurance testing, as needed • Perform CSCI/HWCI integration testing • Perform systems interface testing • Re-test completed SPR actions	System Ready for SQT

7.1 Performance Testing

Testing will be conducted as outlined in the STD and the Interface Requirements Specification (IRS) that will validate the operation(s) of all interface and data exchange parameters. The test cases will provide for processing of predicted user volume and unpredicted or excessive user volume. The Performance Test will also test automated processes routinely initiated by the system or automated processes initiated by users. The automated processes will be tested for normal and excessive volume. The response times of normal and excessive usage will be documented to ensure compliance with SRD specifications.

7.1.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. Participants will be responsible for overseeing the activities to be performed and providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the participants will vary from system to system due to system complexity, size, function, etc.

NOTE: *The person(s) responsible for fulfilling the requirements in this function should not be the persons who performed detailed design or implementation of software in the system. This does not preclude persons who performed detailed design or implementation of software in the system from contributing to the process, for example, by contributing test cases that rely on knowledge of the system internal implementation.*

7.1.2 Entrance Criteria

a. The subject software is under CM.
b. The environment is tested to ensure communications with other systems is reliable.
c. Environment is verified to simulate production environment.
d. Hardware configuration is verified and under CM.
e. STD is complete and ready for execution.
f. Approval of TRR for Integration Test.
g. When necessary, multiple users for stress and enduring testing have been notified.

7.1.3 Inputs

a. Simulated or automated users.
b. Updated STR reflecting CSCI integration results.
c. Test cases, test case procedures, and test data for Performance Testing.
d. Simulated production environment.

7.1.4 Activities

a. Conduct pre-test inspections of target system hardware configurations and interfaces to external systems. Verify that the test environment simulates production environment.
b. Execute test cases in the simulated production environment, and record results on operator logs.
c. Perform stress and endurance testing and record maximum tolerance levels and response times.
d. Upon completion of individual test sessions, debrief test personnel and development staff on test observations.
e. Submit SPRs, as required.

f. Software Development Group addresses SPRs and submits new executable. Re-test until system meets quality gate.

g. Create STR.

7.1.5 Outputs

a. Updated SPR.

b. STR.

c. Software and hardware under CM control.

7.1.6 Exit Criteria

a. Approved STR.

b. Software and hardware ready for SQT.

This page intentionally left blank

8. TEST READINESS REVIEW FOR SYSTEM QUALIFICATION TEST

Before progressing to SQT (i.e., user acceptance testing), a review of the individual components and/or activities necessary for the test will be conducted by project management in the form of a TRR.

PROCESS	TASKS	OUTPUTS
Test Readiness Review	• Verify system software quality and SPR status • Verify baseline STP and STDs • Verify system test environment • Approve testing plan	Authorization to Start System Qualification Test

8.1 Test Readiness Review Checklist

A TRR checklist is recommended for guidance in the assessment of necessary actions and documentation needed to ascertain the status of a system prior to entering the next stage of the SI&T process. Actions and items completed and approved will be noted and the date of completion and approval will be entered on the checklist, while actions and items needing attention will be noted. At the conclusion of the TRR, the checklist will be copied and distributed to all participants. Project management and QA will have final authority for proceeding into the SQT phase.

See Appendix B for the TRR Checklist template.

8.2 Test Readiness Review

The purpose of the TRR meeting is to ensure the quality and status of the system or software prior to its release for SQT. The TRR will be conducted a minimum of two business days before SQT testing begins.

8.2.1 Participants

a. Project management.
b. Customer management.
c. Software development engineers.
d. System test engineers.
e. QA engineers.
f. CM engineers.
g. System network engineers (if necessary).

8.2.2 Entrance Criteria

System has completed the Performance Test phase of the SI&T process.

8.2.3 Inputs

a. Baseline STP and STD(s).
b. Performance Test STR.
c. Testing plan for SQT.
d. Test cases, test case procedures, and test data for SQT.
e. System SPR status report.

8.2.4 Activities

a. Preparation of test schedule.
b. Approval of a test plan.
c. Approval of test cases, test case procedures, and test data.
d. Ascertain status of system.
e. Review of SPR status.
f. Approval of traceability from SRD to test cases.
g. Establish that HWCI is under CM procedures.

8.2.5 Outputs

a. Approval to proceed to SQT phase.
b. Decision to withhold approval of system for SQT phase.

8.2.6 Exit Criteria

Decision regarding the status of system to proceed to SQT phase.

9. SYSTEM QUALIFICATION TEST

SQT is the final SI&T phase prior to user/customer acceptance of the software or system. Experienced users of the system software will work with the system test organization to develop appropriate test cases and test case procedures.

PROCESS	TASKS	OUTPUTS
System Qualification Testing	• Users participate with system test organization in developing/adapting test cases and test case procedures • Users execute test cases • Users participate with system test organization in post test analysis • Preparation of final System Test Report	Final System Test Report Customer acceptance of system

The SQT is intended to verify program performance in accordance with the System or Subsystem Specifications (SSS) and those requirement specifications referenced from the SRS and SRD. This testing includes all functional areas and interfaces to verify the functionality of a totally integrated system. System reliability has been evaluated for independent functional and simultaneous operations and in light and dense usage environments during Performance Testing. All system functions and subsystem interfaces will be independently tested in a systematic manner. System performance will be visually analyzed, augmented by automated data collection, if necessary, and test personnel will record results.

The final results of SQT are documented and included in the final STR. The final STR is then distributed for review and approval. The final STR serves as evidence that the tested software environment has met all requirements as outlined in the SRS and serves the project management as documentation of system acceptance.

9.1 System Qualification Testing Procedures

The testing organization will work with senior users of the proposed system to develop test cases and test case procedures appropriate for SQT. Ideally, the SQT test cases will be adaptations of the previously used test cases and test case procedures. SQT components and objectives will include the following areas:

a. Functional Tests - Functional Tests cover the functional requirements of the SRS.

b. Single User Tests - Single user tests are performed for each of the functional areas to validate the program operation in a one-on-one link.

c. SPR Correction/Closure Tests - These tests are executed to verify fixes to problems and for concurrence with the decision to close the SPR, if necessary.

9.1.1 Participants

a. System users.

b. System test organization.

9.1.2 Entrance Criteria

TRR approval of Performance Testing.

9.1.3 Inputs

a. Software under CM control.

b. Baseline STP and STDs.

c. SDFs for all CSCIs.

d. SPR database information.

e. Approved Performance Test SPR.

f. Approved test cases and test case procedures.

9.1.4 Activities

a. User participation with the system test organization in developing and recording test cases, test case procedures, and data for system testing.

b. User participation with the system test organization in dry run(s) of the test cases, if testing is to be witnessed.

c. User participation in performing the SQT.

d. Revising test cases and test case procedures.

e. Conduct test cases procedures in accordance with the test case. Record test results as they are observed.

f. Re-testing, as needed.

g. Recording of the test results.

h. Analyzing the testing results.

i. Preparation of Final STR.

9.1.5 Outputs

a. SPR against system functions.

b. Updated SDFs.

c. Updated STDs.
d. Final STR.
e. New or updated system software package.

9.1.6 Exit Criteria

a. Project Manager and/or customer acceptance of the system software.
b. Approval of Final STR

This page intentionally blank

GLOSSARY

Aggregate

A mass of distinct things gathered into a total or whole.

Aggregation Level

Effective measurement analysis and reporting requires that the data be aggregated to higher levels of the of the software components and project organizational structure. The aggregation levels define the different ways the measurement data can be grouped and organized for reporting on the project. The aggregation levels describe how the measurement data relates to an existing product and process structures. The organization that allows the measurement results to be combined, and later decomposed, into meaningful pieces of information.

Aggregation Structure

The structure used to define the data according to the defined aggregation levels. The levels may describe the personnel and management structure of the project, or the configuration of physical components of the project. All entries in a structure should be of the same type, such as software modules. However, these entries may reside at various levels of the structure, such as software modules at the unit level, CSCI, or integrated level of the software architecture.

Application

(1.) A complete, self-contained program that performs specific function(s) directly for the user.

(2.) In the TPM process this term refers to one of the two basic measurement activities which comprise the system measurement process. The application activity involves collecting, analyzing, and acting upon the measurement data.
See **Tailoring**

Automated Test Script

A computer readable set of instructions that performs a sequence of steps, sub-steps, or other actions, performed serially, in parallel, or in some combination of consecution, that creates the desired test conditions that the test case is deigned to evaluate.

Baseline

A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures.

Baseline Control

Baseline control is the process that regulates approved and released versions of all software, documentation, and the test environment throughout the test life cycle.

Black Box Testing

This is testing associated with functional testing where the object being tested is treated as a black box. In this type of testing the test object is subjected to inputs and outputs that are verified for conformance to prescribed specifications.

Capacity Testing

Attempts to simulate expected customer peak load operations in order to ensure that the system performance requirements are met. It does not necessarily exercise all of the functional areas of the system, but selects a subset that is easy to replicate in volume. It will ensure that functions which are expected to use the most system resources are adequately represented.

Change Control

The process by which problems and changes to the software, documentation, and test environment are evaluated, approved, rejected, scheduled, and tracked.

Computer Aided Software Engineering (CASE)

A technique for using computers to help with one or more phases of the software life cycle, including the systematic analysis, design, implementation and maintenance of software. Adopting the CASE approach to building and maintaining systems involves software tools and training for the developers who will use them.

Computer Software Configuration Item (CSCI)

An aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process.

Configuration Control

An element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification.

Configuration Item (CI)

Hardware or software, or an aggregate of both, which is designated by the project configuration manager (or contracting agency) for configuration management.

Configuration Management (CM)

A discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements.

Configuration Management Office (CMO)

The Configuration Management Office (CMO) is the officiator of the project life cycle CM process.

Criteria

A standard, rules, or tests by which something can be judged.

Critical Defect

See Criticality

Criticality

The assessment of the impact upon a system of a given error, defect, problem, or discrepancy during the life cycle of a system.

The definition of critical and non-critical system defects or problems should be addressed at a management level and can be different for each system. For any given system error, defect, problem, or discrepancy, an appropriate impact value (i.e., priority) will be assigned.

An example of impact values with the corresponding priority numbers is presented below as contained in IEEE/EIA Std-12207, 1998. The priority that will apply if a problem can result in one or more of these impacts:

PRIORITY	IMPACT
1.	a.) Prevent the accomplishment of an operational or mission essential capability. b.) Jeopardize safety. c.) Cause significant technical, cost, or schedule risks to the project or to life cycle support of the system.
2.	a.) Adversely affect the accomplishment of an operational or mission essential capability and no work-around solution is known. b.) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, and no work-around is known.
3.	a.) Adversely affect the accomplishment of an operational or mission essential capability, but a work-around solution is known. b.) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, but a work-around is known.
4.	a.) Results in user/operator inconvenience or annoyance, but does not affect a required operational or mission essential capability. b.) Results in inconvenience or annoyance for development or support personnel, but does not prevent the accomplishment of the responsibilities of these personnel.
5.	a.) This priority denotes any other effect.

Customer

The organization that procures software systems for itself or another organization.

Developer

An organization that develops software products. The term “develop” may include develop, modification, integration, reengineering, sustaining engineering, maintenance, or any other activity that results in software products. The developer may be a contractor or a government agency.

Discrepancy

An inconsistency or disagreement found during testing between the actual and expected test results.

Document

A data medium and the data recorded on it that generally has permanence and can be read by a human operator or machine. Often used to describe human readable items only (e.g., technical documents, design documents, requirements documents, etc.).

Documentation

(1.) A collection of documents on a given subject.

(2.) The management of documents, that includes the actions of identifying, acquiring, processing, storing, and disseminating.

(3.) Any written or pictorial information describing, defining, specifying, reporting or certifying activities, requirements, procedures, or results.

Driver

A software program that exercises a system or system component by simulating the activity of a higher level component.

Emulation

One system is said to emulate another when it performs in exactly the same way, though perhaps not at the same speed. A typical example would be the emulation of one computer by (a program running on) another. You might use emulation, as a replacement for a system whereas you would use a simulation if you just wanted to analyze it and make predications about it.

Emulator

Hardware or software that performs emulation.

Entry Criteria

A set of decision making guidelines used to determine whether a system under test is ready to move into, or enter, a particular phase of testing. Entry criteria tend to become more rigorous as the test phases progress.

Environment

The infrastructure in which a system is executing, consisting of hardware, operating system software, interfaces, etc.

Exit criteria

A set of decision-making guidelines used to determine whether a system under test is ready to exit a particular phase of testing. When exit criteria are met, either the system under test moves on to the next test phase or the test project is considered complete. Exit criteria tend to become more rigorous as the test phases progress.

Final System Test Report (FSTR)

Used to determine whether system testing is completed and to assure that software is ready for production.

Hardware Configuration Item (HWCI)

An aggregation of hardware that is designated for configuration management and treated as a single entity in the configuration management process.

Independent Verification and Validation (IV&V)

The verification and validation of a software product by an organization that is both technically and managerially separate from the organization responsible for developing the product.

Indicator

A measure or combination of measures that provides insight into a system issue or concept. TPM frequently uses indicators that are comparisons, such as planned versus actual measures. Indicators are generally presented as graphs or tables.

Integration

Combining software or hardware components or both into an overall system.

Integration Testing

The period of time in the software lifecycle during which the application is tested in a simulated production environment to validate the communications and technical architecture of the system. This test phase occurs when all the constituent components of the system under test are being integrated.

Interactive Development Environment (IDE)

A system for supporting the process of writing software. Such a system may include a syntax-directed editor, graphical tools for program entry, and integrated support for compiling and running the program and relating compilation errors back to the source code.

Interface

(1.) A shared boundary (e.g., a hardware component linking two devices or registers, or a portion of storage accessed and/or modified by two or more computer programs).

(2.) To interact or communicate with another system component.

Interface Requirement

A requirement that specifies a hardware, software, or database element with which a system or system component must interface, or that sets forth constraints caused by such an interface.

Interface Specification

A specification that sets forth the interface requirements for a system or system component (e.g., the software interface specification document).

Interface Testing

Tests conducted to ensure that program or system components correctly pass data and/or control to one another.

Issue

An area of concern where obstacles to achieving program objectives might arise. Issues include risks, problems, and lack of information. These three types of issues are defined as:

- Risk -- An area of concern that could occur, but is not certain. A risk is a potential problem. Risks represent the potential for the realization of unwanted, negative consequences from a project event. For example, a project plan may be based on the assumption that a COTS component will be available on a given date. There is a possibility (probability) that the COTS may be delayed and have some amount of negative impact on the project.
- Problem -- An area of concern that a project is currently experiencing or is relatively certain to experience. For example, a shortage of staff with the right skills may be an actual problem that is delaying the project.
- Lack of Information -- An area where the available information is inadequate to reliably predict project impact. Thus, satisfaction of project objectives is questionable even if no problems or risks are present. For example, lack of information about the size of the software to be developed could result in the project "discovering" that it has more work to do than originally planned.

Measure

The result of counting or otherwise quantifying characteristics of a process or product. Measures are numerical values assigned to system attributes according to defined criteria.

Measured (or actual) Value

Actual, current measurement data, such as hours of effort expended or line of code produced.

Measurement

The process of assigning quantitative values of system properties, according to some defined criteria. This process can be based on estimation or direct measurement. Estimation defines planned or expected measures. Direct measurement results in actual measures.

Measurement Analysis

The uses of measurement data to identify problems, assess problem impact, project an outcome, or evaluate alternatives related to system issues.

Measurement Analyst

The person(s) or team responsible for tailoring and applying system measures for a given project or task.

Measurement Information

Knowledge derived from analysis of measurement data and measurement indicators.

Milestone

A scheduled event for which some project or task member or manager is held accountable. A milestone is often used to measure progress.

Module

A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading.

Note: *The terms 'module', 'component', and 'unit' are often used interchangeably or defined to be sub-elements of one another in different ways depending on the context.*

Non-Critical Defect

See Criticality

Performance Testing

The period of time in the system or software development lifecycle during which the response times for the application are validated to be acceptable. The tests ensure that the system environment will support production volumes, both batch and on-line.

Priority

A measure of the elements of importance related to the repair of a system problem that are not considered in defining the severity of a system problem.

Project Manager (PM)

The official responsible for acquiring, developing, or supporting a system to meet technical, cost, schedule, and quality requirements. Acquisition, development, and support will include both internal tasks and work that is contracted to another source.

Quality Assurance (QA)

A planned and systematic pattern of all actions necessary to provide adequate confidence that the product optimally fulfils customers expectations.

Quality Control (QC)

The assessment of product compliance. Independently finding deficiencies assures compliance of the product with stated requirements.

Requirement

(1.) A condition or capability needed to solve a problem or achieve an objective.

(2.) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document. The set of all requirements forms the basis of development.

Regression testing

Part of the test phase of software development where, as new modules are integrated into the system and the added functionality is tested, previously tested functionality is re-tested to assure that no new module has corrupted the system.

Risk

An area of concern that may occur, but is not certain. A risk is a potential problem. Risks represent the potential for the realization of unwanted, negative consequences from a project event. For example, a project plan may be based on the assumption that a commercial off the

shelf (COTS) component will be available on a given date. There is a possibility (probability) that the COTS may be delayed and have some amount of negative impact on the project.

Severity

The degree to which a problem adversely influences the system's operation or the overall test effort.

Simulation

Attempting to predict aspects of the behavior of a system by creating an approximate (mathematical) model of it. This can be done by physical modeling, by writing a special-purpose computer program or using a more general simulation package, aimed at a particular kind of simulation. Typical examples are aircraft simulators or electronic circuit simulators.

Simulator

Hardware or software that performs simulation.

Software Design Specification (SDS)

A document that records the design of a system or system component; typical contents include: system and/or component algorithms, control logic, data structures, data set use, input/output formats, and interface descriptions.

Software Development File (SDF)

The developer shall document the development of each Computer Software Unit (CSU), Computer Software Component (CSC), and CSCI in Software Development Files (SDF). The developer shall establish a separate SDF for each CSU or a logically related group of CSUs, for each CSC or a logically related group of CSCs, and for each CSCI. The developer shall document and implement procedures to establish and maintain SDFs. SDFs may be generated, maintained, and controlled by automated means. To reduce duplication, SDFs should not contain information provided in other documents or SDFs. The set of SDFs shall include (directly or by reference) the following information:

- Design considerations and constraints.
- Design documentation and data.
- Scheduling and status information.
- Test requirements and responsibilities.
- Test case, test case procedures, and results.

Software Life Cycle

The phases a software product goes through between when it is conceived and when it is no longer available for use. The software life cycle typically includes the following: requirements, analysis, design, construction, testing (validation), installation, operation, maintenance, and retirement. The development process tends to run iteratively through these phases rather than linearly; several models (spirals, waterfall, etc.) have been proposed to describe this process. Other processes associated with a software product are: quality assurance, marketing, sales, and support.

Software Management Plan

A project plan for the development of the software component of a system or for the development of a software product.

Software Requirements Document (SRD)

This is a formal document derived from the Software Requirements Specification (SRS) that sets forth the requirements, specifications, and standards for a system (e.g., a software product). Typically included are functional specifications and requirements, performance specifications and requirements, interface specifications and requirements, design specifications and requirements, and development requirements and standards.

Software Requirements Specification (SRS)

A specification that sets forth the requirements for a system component; (e.g., a software product). Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards.

Software Tool

Computer programs used to help develop, test, analyze, or maintain another computer program or its documentation.

Specification

Documentation containing a precise, detailed, verifiable description of particulars with respect to the requirements, design, function, behavior, construction, or other characteristics of a system or system component.

Stub

(1.) A dummy procedure used when linking a program with a run-time library. The stub routine need not contain any code and is only present to prevent "undefined label" errors at link time.

(2.) A local procedure in a remote procedure call (RPC). The client calls the stub to perform some task and need not necessarily be aware that RPC is involved. The stub transmits parameters over the network to the server and returns the results to the client/caller.

System

(1.) Any large program.

(2.) The entire computer system, including the input/output devices, supervisor program or operating system and possibly other software.

System Problem Report (SPR)

A form that is used to record a discrepancy discovered during the Integration Test, Performance Test and System Qualification Test phases of the SI&T process concerning a Computer Software Configuration Item, a software system or subsystem, other software related items, and associated documentation.

System Problem Report (SPR) Status Report

The System Problem Report Status Report is used during the SPR Status Review to determine if the SPRs are being processed appropriately and expeditiously.

System Testing

The period of time in the software lifecycle during which the implementation of each requirement is validated.

Tailoring

In the TPM process, this term refers to one of the two basic measurement activities, which comprise the system measurement process. The tailoring activity includes identification and prioritization of program issues, selection and specification of appropriate system measures, and integration of the measurement requirements to the developer's system process.
See **Application**.

Test

The process of exercising a product to identify differences between expected and actual behavior.

Test Artifacts

An item created during the system integration and test process that is preserved upon completion of the test process (e.g., test plans, requirements documentation, automated test scripts, and test documentation).

Test Case

A description of a test to be executed for or focused on a specific test aim.

Test Case Procedures

A sequence of steps, sub-steps, and other actions, performed serially, in parallel, or in some combination of consecution, that creates the desired test conditions that the test case is designed to evaluate.

Test Case (Setup) Suite

The steps required to configure the test environment for execution of a test case.

Testing Condition

System state or circumstance created by proceeding through some combination of steps, sub-steps, or actions in a test case.

Testing Environment

The infrastructure in which the test is performed, consisting of hardware, system software, test tools, and procedures.

Test Plan

In a test plan the general structure and the strategic choices with respect to the test to be executed are formulated. The test plan forms the scope of reference during execution of the test and also serves as an instrument to communicate with the customer of the test. The test plan is a

description of the test project, including a description of the activities and planning, therefore it is *not* a description of the tests themselves.

Test Readiness Review (TRR)

Review conducted to determine whether a software test phase has been completed and to assure that the software is prepared for the next step in the formal integration and testing procedures. Software test procedures and results are evaluated, for compliance with the software testing requirements and system descriptions, for adequacy in accomplishing testing goals. Also, provides the forum for updating and revising operational and supporting documentation.

Test Resources

Aids that are used by a test tool for collecting, tracking and controlling information. This information is:

- Software requirements defined in the Software Requirements Document.
- Test requirements defined in the System Test Description.
- Automated test case scripts as defined in the System Test Description.
- SPRs as determined at each phase of the System Integration and Testing process.

This information is controlled by Configuration Management at the end of the SI&T process for use whenever further testing may be conducted, using a testing tool, during the remaining lifecycle of the software or system.

Test Tools

The software, hardware, systems, or other instruments that are used to measure and test an item.

Traceability

Degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor, successor, or master-subordinate relationship to one another (e.g., the degree to which the requirements and design of a given software component match).

Unit

The lowest element of a software hierarchy that contains one or more of the following characteristics:

(1) A unit comprising one or more logical functional entities.

(2) An element specified in the design of a computer software component that is separately testable.

(3) The lowest level to which software requirements can be traced.

(4) The design and coding of any unit can be accomplished by a single individual within the assigned schedule.

Unit Test

The process of ensuring that the unit executes as intended. This usually involves testing all statements and branch possibilities.

Version

One of a sequence of copies of a system, each incorporates new modifications.

Version Identifier

A unique identifier assigned to baseline software, documentation, and test environment.

Version Control

The process by which all changes to the software, documentation, and test environment are compiled and built into a new version of the system.

Version Control Report

A report that details all changes and enhancements made to current version of the software, documentation, and test environment.

White Box Testing

This type of testing is associated with structural testing in which the testing can be characterized as being tied to implementation details, such as control methods, database design, coding details, and logic paths. The process of how an individual input is treated to produce a given output is ascertained. Structural testing is sometimes referred to as "clear box testing" since white boxes are considered opaque and do not really permit visibility into the code.

Work Breakdown Structure (WBS)

A work breakdown structure for software defines the software-related elements associated with program work, work activities, and products. Many measures are aggregated and analyzed at various WBS levels.

BIBLIOGRAPHY

Black, Rex. *Managing The Testing Process,* Redman, WA: Microsoft Press, 1999

Koomen, Tim and Pol, Martin. *Test Process Improvement*, Essex, England, UK: Pearson Education Limited, 1999

Evans, Michael and Marciniak, John. *Software Quality Assurance and Management,* New York, NY: John Wiley & Sons, 1987

Carnegie Mellon University, Software Engineering Institute. *The Capability Maturity Model: Guidelines for Improving the Software Process*, 1995.

Institute of Electrical and Electronics Engineers (IEEE). "Glossary of Software Engineering Terminology", IEEE-Std-610.12, 1990.

Institute of Electrical and Electronics Engineers (IEEE)/Electronic Industries Alliance (EIA). "Software Life Cycle Processes", IEEE/EIA Std-12207, 1998.

U.S. Department of Education. "SFA System Integration & Testing Approach, SFA Modernization", Undated.

Federal Systems Integration and Management Center (FEDSIM). "FEDSIM Writers Guide, Version 2", May 1994

Free On-Line Dictionary Of Computing web site at www.foldoc.org

This page intentionally left blank

APPENDIX A

TESTING TECHNIQUES

This page intentionally left blank

Testing Techniques

Software managers and test managers must be familiar with the various techniques employed in system testing. Application of testing techniques and the degree of testing varies from project to project. Software test planning should identify the following items:

a. The specific techniques to be employed.

b. When in the testing process the techniques will be used.

c. To what degree the tests should validate the overall system requirements.

Testing should be differentiated from the notion of debugging. The characterizations of each are presented in Table A-1.

TESTING	DEBUGGING
Uses known condition, predefined test case procedures and has a predictable outcome. The only unpredictable outcome is whether or not a program passes the test.	Starts from an unknown initial condition and does not have a predictable outcome.
Tests are designed and scheduled, and they are predictable, constrained, and formal by nature.	Intuitive and experimental and requiring detailed design knowledge, and freedom to perform deductive analysis.
A demonstration of correctness or error.	A programmer's vindication.

Table A-1 Testing Versus Debugging

Understanding that testing is a formal discipline, one can approach the task from the two ends of the testing spectrum. At one end of the spectrum is the concept of structural testing. Structural testing can be characterized as being tied to implementation details, such as control methods, database design, coding details, and logic paths. This form of testing is often referred to as "White Box" testing. SU testing tends to be structural in nature.

At the other end of the spectrum is the concept associated with functional testing where the object under test is treated as a "Black Box." In this strategy, the test object is subjected to inputs and resulting outputs are verified for conformance to a prescribed specification. All other testing conducted during the SI&T process tends to be functional in nature.

In developing a strategy for testing at each level of system development, a combination of structural and functional tests is recommended. Table A-2 recommends the application of the structural and functional techniques through system development. The basic techniques are listed below:

a. Test Path Coverage Analysis is a structural technique easily applied during the SU test phase. Path testing involves exercising selected logic paths through the control structure of the software. Algorithms that determine path selection and the generated outputs of a path are also validated using this methodology. Anything less than complete coverage means that untested code is being integrated into the system.

b. Equivalence Partitioning is a strategy employed in developing functional tests. Developing a set of test cases that incorporates all possible combinations of inputs to the system is typically not possible due to time, cost, and resource constraints. Equivalence partitioning involves developing tests that incorporate a subset of the inputs that exercise the maximum number of paths and algorithms in the system software. For example, you do not need to generate a test for all the ways to cause a specific result, but you must generate at least one test that will exercise that system to cause that specific result.

c. Boundary Condition Analysis employs testing boundary conditions in either a structural or functional format. This form of testing involves identifying and using input values that exercise the maximum and minimum boundaries by input of test values just above and below the specified range end points. The intent is to validate system input/output tolerances and associated algorithms. Tests can be developed based on the user perspective (functional) or from a simulation of peripheral inputs (structural).

d. Input Syntax Validation is a functional technique used to validate the man-machine interface to the system. This form of testing involves exercising the man-machine interface to ensure acceptance of valid inputs to the system and the response to invalid inputs as to direct and/or indirect side effects.

Note in Table A-2 that, the underlying strategy moves from a predominantly structural form to a more functional form as the system moves through the phases of the System Integration and Testing process. The table is not all inclusive and the reader is invited to become educated on detailed issues of software testing prior to developing a test plan, test cases, or test case procedures or performing software testing.

Table A-2			
Test Phase	**Test Level**	**Technique**	**Purpose**
SU Test Phase	Unit Test and Unit Integration Test	Structural Path Testing	Static testing of logic branches to validate control structures, loop boundaries, and error recovery processing.
		Structural Boundary Condition Testing	Static testing of input and output parameter tolerances and accuracy of algorithm implementations.
		Functional user input syntax validation.	Verifies processing of user input data to ensure proper conversation to internal form and validate error message generation in response to invalid inputs.
	CSCI Qualification Test	Functional test bed based on equivalence partitioning.	Equivalence partitioning involves defining the minimum number of functional test to exercise the maximum number of intra-CSCI logic paths and data interfaces. Verifies CSCI functional capability against the SRD and STP.
		Functional user input syntax validation.	Valid and invalid inputs to uncover errors in the user interfaces. Verify error-handling facilities as stated in the SRD.
		Structural Boundary Condition testing.	Testing the intra-CSCI and peripheral interfaces to find errors in input and output data tolerances and verify CSCI data processing and limits are correctly implemented.

Table A-2			
Integration Test Phase	Integration Test	Functional test bed based on equivalence partitioning.	Test inter-CSCI logic paths and data interfaces. Verifies system software functional capability against the SRD. Testing based on defined test case(s) and test case procedure(s) documented in the STD.
		Functional user input syntax validation.	Valid and invalid inputs to uncover errors in the user interfaces. Verify error-handling facilities as stated in the SRD. Testing based on defined test case(s) and test case procedure(s) documented in the STD.
		Structural Boundary Condition testing.	Testing of inter-CSCI and peripheral interfaces to find errors in input and output data tolerances and verify that the systems data processing and limits are correctly implemented. Testing based on defined test case(s) and test case procedure(s) documented in the STD.
	Regression Test	Regression Test configuration and environment.	Functional and/or structural tests developed specifically to test reported high-priority problems and/or critical algorithms. Used to verify the integrity of the production program after changes are made to the software. Testing based on defined test case(s) and test case procedure(s) documented in the STD.

Table A-2			
Performance Test Phase	Performance Test	Functional test configuration and environment.	High volume functional test bed execution to determine system software load capacity and ability to meet overall requirements stated in the SRD. Testing based on defined test case(s) and test case procedure(s) documented in the STD.
System Qualification Test Phase	System Qualification Test	Functional test configuration and environment.	Functional tests developed to test the system in a production environment, as a customer/user would utilize the system. Testing based on defined test case(s) and test case procedure(s) documented in the STD.

This page intentionally left blank

APPENDIX B

TEST READINESS REVIEW CHECKLIST TEMPLATE

This page intentionally left blank

SI&T TEST READINESS REVIEW CHECKLIST

Preparer: _____ Date of TRR: _____

System/Project Identification: _____

Testing phase: Integration Testing ☐

Performance Testing ☐

System Qualification Testing ☐

SI&T TEST READINESS REVIEW CHECKLIST	
Criterion	**(Yes/No/Not Applicable), Date of Entry, Record References To and Reasons For Non-Compliant Items**
System Test Plan (STP) submitted and approved.	
System Test Description (STD) submitted and approved.	
Software Unit (SU) Software Development File (SDF) submitted and approved.	
Appropriate System Test Report (STR) submitted and approved.	
Appropriate testing environment documented, approved and established.	
Appropriate test configuration documented, approved and established.	
Overall testing plan for testing phase submitted and approved.	
Test cases, test case procedures, and appropriate test data for system Integration Test submitted and approved.	
Test cases, test case, and appropriate test data procedures for system Performance Test submitted and approved.	

SI&T TEST READINESS REVIEW CHECKLIST	
Criterion	**(Yes/No/Not Applicable), Date of Entry, Record References To and Reasons For Non-Compliant Items**
Test cases, test case procedures, and appropriate test data for System Qualification Test (SQT) submitted and approved.	
Appropriate test cases, test case procedures, and necessary test data placed under Configuration Management (CM) procedures.	
Traceability from Software Requirements Document (SRD) to Test Case(s) established.	
Computer Software Configuration Item (CSCI) modules under Configuration Management (CM) procedures.	
Hardware Configuration Item (HWCI) under Configuration Management (CM) procedures.	
Commercial Off the Shelf (COTS) software under Configuration Management (CM) procedures.	
Applicable deviations/waivers regarding testing function have been submitted and approved.	
Testing schedule prepared, distributed, and approved by all appropriate personnel and organizations.	
System Report (SPR) forms documented, inventoried, and their status approved.	
Prior milestones completed (i.e., all prior exit criteria satisfied).	
Needed security requirements satisfied, documented and approved.	
Entrance criteria for the next appropriate testing phase established and approved.	
Exit criteria and necessary output(s) for the next appropriate testing phase established and approved.	

APPENDIX C

SYSTEM TEST REPORT TEMPLATE

This page intentionally left blank

APPENDIX D

SYSTEM PROBLEM REPORT TEMPLATE

This page intentionally left blank